	Type	Hit	S arch Text	DB
1	BRS	17793	hot adj2 sp t or h t adj2 (addr ss r addr sses r addressing) sam (thrash or thrashing r thrashed)	USPAT; US-PGPUB; EP ; JP ; IBM_TDB
2	BRS	17793	hot adj2 spot or hot adj2 (address or addresses or addressing) with (thrash or thrashing or thrashed)	USPAT; US-PGPUB; EPO; JPO; IBM_TDB
3	BRS	5	(hot adj2 spot or hot adj2 (address or addresses or addressing)) with (thrash or thrashing or thrashed)	USPAT; US-PGPUB; EPO; JPO; IBM_TDB
4	BRS	9	(hot adj2 spot or hot adj2 (address or addresses or addressing)) same (thrash or thrashing or thrashed)	USPAT; US-PGPUB; EPO; JPO; IBM_TDB
5	BRS	10	(hot adj2 spot or hot adj2 (address or addresses or addressing)) with (cache or caching or cached) with (set or frame)	USPAT; US-PGPUB; EPO; JPO; IBM_TDB
6	BRS	36	(detect or detecting or detection) with (thrash or thrashing or thrashed)	USPAT; US-PGPUB; EPO; JPO; IBM_TDB
7	BRS	16	(detect or detecting or detection) with (thrash or thrashing or thrashed) same (cache or caching or cached)	USPAT; US-PGPUB; EPO; JPO; IBM_TDB

	Tim Stamp
1	2004/01/06 11:13
2	2004/01/06 11:13
3	2004/01/06 11:13
4	2004/01/06 11:16
5	2004/01/06 11:31
6	2004/01/06 11:32
7	2004/01/06 11:32

US-PAT-NO: 6625695

DOCUMENT-IDENTIFIER: US 6625695 B2

TITLE: Cache line replacement policy enhancement to avoid
memory page thrashing

DATE-ISSUED: September 23, 2003

US-CL-CURRENT: 711/134, 711/133

APPL-NO: 10/ 319738

DATE FILED: December 13, 2002

PARENT-CASE:

This patent application is a Continuation of U.S. patent application Ser.
No. 09/675,765, entitled "Cache Line Replacement Policy Enhancement To Avoid
Memory Page Thrashing", filed Sep. 29, 2000 now U.S. Pat. No. 6,523,092.

---------- KWIC ---------

Detailed Description Text - DETX (24):

When a pathological pattern, such as a stream of writes or the data accesses
is applied to a processor cache and chipset memory controller

c mbination, the
r gularity of the acc sses combined with the LRU lin allocation
p licy can
cause worst case memory behavior. Many of these patt rns can be detected and
defeated through a modification to the cache's LRU policy. One such
modification detects likely page-thrashing conditions between reads and
writeback data, and modifies the line allocation policy in response.

Detailed Description Text - DETX (27):

In order to detect the thrash condition, the cache controller requires
information about the paging behavior of the memory subsystem. Sometimes this
information can be complex, as in the case where several different memory
technologies are populated in the same platform. The present invention
provides a mask that the cache controller can use to drive its decisions about
page conflicts.

Detailed Description Text - DETX (28):

FIG. 4 is a schematic diagram of a thrash condition detector. Cache
controller 400 receives input signals THRASH BIT MASK 402 and REQUEST ADDRESS
404. THRASH BIT MASK 402 is a set of address mask signals coupled to AND gate
410 to detect the thrash condition. The mask 402 ensures that only the
necessary bits of the addresses are tested. The particular

PGPUB-DOCUMENT-NUMBER: 20020099912

PGPUB-FILING-TYPE: new

DOCUMENT-IDENTIFIER: US 20020099912 A1

TITLE: Memory system

PUBLICATION-DATE: July 25, 2002

US-CL-CURRENT: 711/119

APPL-NO: 09/ 923339

DATE FILED: August 8, 2001

FOREIGN-APPL-PRIORITY-DATA:

COUNTRY	APPL-NO	DOC-ID	APPL-DATE
JP	2001-012608	2001JP-2001-012608	January 22, 2001

--------- KWIC ---------

Summary of Invention Paragraph - BSTX (9):

[0008] While thrashing may be avoided by adjusting the address of data by inserting dummy data in a sequence of data, it is not easy to detect the occurrence of thrashing and specify the location where thrashing occurs. Thrashing may be prevented by designing the cache memory in a

full associative
syst m. But, the c mplexity f ch cking a hit in th cache
m mory inevitably
nlarges th hardware, thus increasing the time needed to access the cache and
decreasing the mounting efficiency. Because of this disadvantages, the full
associative system is not generally employed.

Summary of Invention Paragraph - BSTX (12):

[0010] According to the invention, means for detecting the occurrence of
thrashing is provided between the cache memory and the main memory in order to
avoid thrashing without lowering the speed of accessing the cache memory or the
mounting efficiency. Another feature of the invention lies in that means for
storing thrashing data is provided to suppress the thrashing-originated
reduction in the execution speed of a processor.

Detail Description Paragraph - DETX (7):

[0021] The invention has a cache memory of the direct mapping system or
n-way set associative system as a level 1 cache memory and detects thrashing
which is the overflow of data from the cache memory when a specific set in the
cache memory is used within the time in which there are a greater number of
pieces of line data than the associativity of the cache memory. The invention
also has a mechanism which reduces thrashing to be detected.

US-PAT-NO: 5752261

DOCUMENT-IDENTIFIER: US 5752261 A

TITLE: Method and apparatus for detecting thrashing in a cache memory

DATE-ISSUED: May 12, 1998

US-CL-CURRENT: 711/133, 711/128

APPL-NO: 08/ 745035

DATE FILED: November 7, 1996

--------- KWIC --------

TITLE - TI (1):
Method and apparatus for detecting thrashing in a cache memory

Brief Summary Text - BSTX (2):
The present invention relates generally to a cache memory, and more particularly to a method and apparatus for improving the performance of a cache memory by detecting and reducing thrashing in the cache memory.

Bri f Summary T xt - BSTX (9):

What is ne ded theref re is a method and apparatus f r improving the performance of a direct-mapped cache memory and a set-associative cache memory by detecting which memory references are causing thrashing therein.

Brief Summary Text - BSTX (12):

In accordance with one embodiment of the present invention, there is provided a method of detecting thrashing in a cache memory having a plurality of cache lines. The method includes the steps of storing a first data in a first main memory location identified by a first address having a first page index and a first tag; storing a second data in a second main memory location identified by a second address having the first page index and a second tag; storing the first data in a first cache line identified by the first page index; referencing the cache memory with the second address; replacing the first data stored in the first cache line with the second data; storing the first tag in a third memory location; and storing the second tag in a fourth memory location.

Brief Summary Text - BSTX (15):

It is an object of the present invention to provide a new and

us ful m th d
f d te ting thra hing in a cach m m ry.

Brief Summary Text - BSTX (16):
It is also an object of the present invention to provide an improved method
of detecting thrashing in a cache memory.

Brief Summary Text - BSTX (17):
It is another object of the present invention to provide a new and useful
method to detect cache thrashing in computer systems employing direct mapped
cache memory.

Brief Summary Text - BSTX (21):
It is yet another object of the present invention to provide a cache
controller that detects and identifies thrashing pages of main memory.

Drawing Description Text - DRTX (6):
FIG. 5 is a simplified block diagram showing the thrashing detection and
reduction circuit of the cache controller of FIG. 1: and

Detailed Description Text - DETX (24):
If the computer system 10 could detect which physical pages 37 of the main
memory 20 were causing the cache 22 to thrash, then the CPU 12 could remedy the

situati n by (1) m ving the thra hing physical pag s 37 s that they hav
diff r nt valu s f r the page index fi ld 90 and (2) updating the translation
table stored in the translation buffer 26 and the main memory 20 so that the
virtual pages 35 corresponding to the thrashing physical pages 37 map correctly
to the moved physical pages 37. By doing so, the previously thrashing physical
pages 37 will no longer map to the same location in cache 22 and as a result
will no longer thrash with one another.

Detailed Description Text - DETX (26):

Referring now to FIG. 5, there is shown a thrashing detection and reduction
circuit (TDRC) 100 of the cache controller 16 (FIG. 1). The TDRC 100 includes
a page index monitor (PIM) 102 and a page index tracker (PIT) 120 which
collectively determine physical pages 37 which are causing the cache 22 to
thrash. The PIM 102 includes a page index monitor controller (PIMC) 104 and a
page index monitor memory (PIMM) 106.

Detailed Description Text - DETX (32):

Referring now to FIG. 6, there is shown a procedural flowchart 300 setting
forth the operation of the thrashing detection and reduction circuit (TDRC) 100
(FIG. 5) after the cache 22 has been initialized. The operation of the TDRC

100 is initiat d (step 310) as a r sult of th PIMC 104 r ceiving an cache
access signal from the stat signal g nerat r 68 (FIG. 4) n lin 72. Up n
receiving the cache access signal from the state signal generator 68, the PIMC
104 increments the access count field 112 identified by the page index field 90
(step 320). The PIMC 104 also increments the replacement count field 110
identified by the page index field 90 if the PIMC 104 also received a cache
replacement signal from the state signal generator 68 on line 74.

Detailed Description Text - DETX (41):

Once the CPU 12 has determined which physical pages 37 are thrashing, the
CPU 12 may reduce thrashing between the detecting physical pages 37 by moving
the thrashing physical pages 37 in physical memory 20 such that they have
different values for the page index field 90. Referring back to FIG. 2 for a
situation in which the CPU 12 has determined that a physical page 37A is
thrashing with a physical page 37B, the CPU 12 may obtain the above result by
(1) mapping an unallocated virtual page 35C to the physical page 37B, (2)
mapping the virtual page 35B originally associated with the physical page 37B
to a physical page 37C having a different value for the page index field 90,
and (3) copying the data from the physical page 37B to the physical page 37C.

By mapping in this manner, the pr vi usly thrashing physical pages 37A and 37B n w ar r spectively stor d in physical pag s 37A and 37C which do not map to the same location in the cache 22 and as a result will not thrash with one another.

Claims Text - CLTX (1):

1. A method of detecting thrashing in a cache memory having a plurality of cache lines, comprising the steps of:

US-PAT-NO: 5630097

DOCUMENT-IDENTIFIER: US 5630097 A

TITLE: Enhanced cache operation with remapping of pages for optimizing data relocation from addresses causing cache misses

DATE-ISSUED: May 13, 1997

US-CL-CURRENT: 711/165, 710/33 , 711/130 , 711/133

APPL-NO: 08/ 178487

DATE FILED: January 7, 1994

PARENT-CASE:

This application is a continuation of application Ser. No. 07/716,207 filed Jun. 17, 1991, now abandoned.

---------- KWIC ---------

Abstract Text - ABTX (1):

A computer system executing virtual memory management and having a cache is operated in a manner to reduce cache misses by remapping pages of physical

m m ry fr m which cach miss s ar detected. Th m th d includ s detecting
cach misses, as by obs rving a h fill perati ns n th syst m bus, and then
remapping the pages in the main memory which contain the addresses of the most
frequent cache misses, so that memory references causing thrashing can then
coexist in different pages of the cache. For a CPU executing a virtual memory
operating system, a page of data or instructions can be moved to a different
physical page frame but remain at the same virtual address, by simply updating
the page-mapping tables to reflect the new physical location of the page, and
copying the data from the old page frame to the new one. The primary feature
of the invention is to add bus activity sampling logic to the CPU and enhance
the operating system to allow the operating system to detect when cache
thrashing is occurring and remap data pages to new physical memory locations to
eliminate the cache thrashing situation.

Brief Summary Text - BSTX (9):

The process of assigning virtual memory pages to physical memory pages also
defines the subset of CPU cache locations where this data can reside, as cache
locations are selected as a function of physical page address, using a cache
tag which is part of the physical address. Current operating systems have no

m chanism f r fav ring the selecti n f ne physical page v r an ther to
reduc the lik lih d of cache thra hing, and n m chanism t detect if a group
of physical pages are cache thrashing and remap one or more of the thrashing
pages to new physical pages to alleviate the problem.

Brief Summary Text - BSTX (12):

In accordance with one embodiment of the invention, a computer system
executing virtual memory management and having a cache is operated in a manner
to reduce cache misses by remapping pages of physical memory from which cache
misses are detected. In a CPU executing a virtual memory type of operating
system, data is handled in pages and the cache is accessed via physical
addresses. The method includes detecting cache misses, as by observing cache
fill operations on the system bus, and then remapping the pages in the main
memory which contain the addresses of the most frequent cache misses, so that
memory references causing thrashing can then coexist in different pages of the
cache. Two memory addresses which are in different physical page frames but
which map to the same location in the cache may be arbitrarily placed in the
direct-mapped cache at the same cache page; alternate reference to these two
addresses by a task executing on the CPU would cause thrashing. However, if

the l cation f n of these address s in main m m ry is changed, th data it ms having th s addr ss s can th n c exist in th cache, and p rformance will be markedly improved because thrashing will no longer result. For a CPU executing a virtual memory operating system, a page of data or instructions can be moved to a different physical page frame but remain at the same virtual address. This is accomplished by simply updating the page-mapping tables to reflect the new physical location of the page, and copying the data from the old page frame to the new one. In brief, the primary feature of the invention is to add bus activity sampling logic to the CPU and enhance the operating system to allow the operating system to detect when cache thrashing is occurring and remap data pages to new physical memory locations to eliminate the cache thrashing situation.

US-PAT-NO: 6154812

DOCUMENT-IDENTIFIER: US 6154812 A

TITLE: Method for inhibiting thrashing in a multi-level non-blocking cache system

DATE-ISSUED: November 28, 2000

US-CL-CURRENT: 711/122, 711/140 , 711/167

APPL-NO: 08/ 881725

DATE FILED: June 25, 1997

PARENT-CASE:

CROSS-REFERENCES TO RELATED APPLICATIONS

The subject matter of the present application is related to that of co-pending U.S. patent application Ser. No. 08/881,958 identified as Docket No. P2345/37178.830071.000 for AN APPARATUS FOR HANDLING ALIASED FLOATING-POINT REGISTERS IN AN OUT-OF-ORDER PROCESSOR filed concurrently herewith by Ramesh Panwar; Ser. No. 08/881,729 identified as Docket No. P2346/37178.830072.000 for APPARATUS FOR PRECISE ARCHITECTURAL UPDATE IN AN OUT-OF-ORDER PROCESSOR filed concurrently herewith by Ramesh Panwar and Arjun Prabhu; Ser. No. 08/881,726 identified as Docket No. P2348/37178.830073.000 for

**AN APPARATUS FOR
NON-INTRUSIVE CACHE FILLS AND HANDLIN OF L AD MISSES filed concurrently
herewith by Ramesh Panwar and Ricky C. Hetherington; Ser. N . 08/881,908
identified as Docket No. P2349/37178.830074.000 for AN APPARATUS FOR HANDLING
COMPLEX INSTRUCTIONS IN AN OUT-OF-ORDER PROCESSOR filed concurrently herewith
by Ramesh Panwar and Dani Y. Dakhil; Ser. No. 08/882,173 identified as Docket
No. P2350/37178.830075.000 for AN APPARATUS FOR ENFORCING TRUE DEPENDENCIES IN
AN OUT-OF-ORDER PROCESSOR filed concurrently herewith by Ramesh Panwar and Dani
Y. Dakhil; Ser. No. 08/881,145 identified as Docket No. P2351/37178.830076.000
for APPARATUS FOR DYNAMICALLY RECONFIGURING A PROCESSOR filed concurrently
herewith by Ramesh Panwar and Ricky C. Hetherington; Ser. No. 08/881,239
identified as Docket No. P2518/37178.830095.000 for A METHOD FOR ENSURING
FAIRNESS OF SHARED EXECUTION RESOURCES AMONGST MULTIPLE PROCESSES EXECUTING ON
A SINGLE PROCESSOR filed concurrently herewith by Ramesh Panwar and Joseph I.
Chamdani; Ser. No. 08/882,175 identified as Docket No. P2355/37178.830078.000
for SYSTEM FOR EFFICIENT IMPLEMENTATION OF MULTI-PORTED LOGIC FIFO STRUCTURES
IN A PROCESSOR filed concurrently herewith by Ramesh Panwar; Ser. No.
08/882,311 identified as Docket No. P2365/37178.830080.000 for AN APPARATUS FOR
MAINTAINING PROGRAM CORRECTNESS WHILE ALLOWING LOADS**

TO BE BO STED PAST ST RES IN AN UT-OF-ORDER MACHINE fil d c ncurrently her with by Ram sh Panwar, P. K. Chidambaran and Ricky C. Hetheringt n; Ser. No. 08/881,731 identified as Docket No. P2369/37178.830081.000 for APPARATUS FOR TRACKING PIPELINE RESOURCES IN A SUPERSCALAR PROCESSOR filed concurrently herewith by Ramesh Panwar; Ser. No. 08/882,525 identified as Docket No. P2370/37178.830082.000 for AN APPARATUS FOR RESTRAINING OVER-EAGER LOAD BOOSTING IN AN OUT-OF-ORDER MACHINE filed concurrently herewith by Ramesh Panwar and Ricky C. Hetherington; Ser. No. 08/882,220 identified as Docket No. P2371/37178.830083.000 for AN APPARATUS FOR HANDLING REGISTER WINDOWS IN AN OUT-OF-ORDER PROCESSOR filed concurrently herewith by Ramesh Panwar and Dani Y. Dakhil; Ser. No. 08/881,847 identified as Docket No. P2372/37178.830084.000 for AN APPARATUS FOR DELIVERING PRECISE TRAPS AND INTERRUPTS IN AN OUT-OF-ORDER PROCESSOR filed concurrently herewith by Ramesh Panwar; Ser. No. 08/881,728 identified as Docket No. E-2398/37178.830085.000 for NON-BLOCKING HIERARCHICAL CACHE THROTTLE filed concurrently herewith by Ricky C. Hetherington and Thomas M. Wicki; Ser. No. 08/881,727 identified as Docket No. P2406/37178.830086.000 for NON-THRASHABLE NON-BLOCKING HIERARCHICAL CACHE filed concurrently herewith by Ricky C. Hetherington, Sharad Mehrotra and Ramesh Panwar; Ser. No. 08/881,065

id ntifi d as Docket N . P2408/37178.830087.000 for IN-LINE BANK CONFLICT
DETECTION AND RESOLUTION IN A MULTI-P RTED NON-BLOCKIN CACHE filed
concurrently herewith by Ricky C. Hetherington, Sharad Mehrotra and Ramesh
Panwar; and Ser. No. 08/882,613 identified as Docket No. P2434/37178.830088.000 for SYSTEM FOR THERMAL OVERLOAD DETECTION AND PREVENTION
FOR AN INTEGRATED CIRCUIT PROCESSOR filed concurrently herewith by Ricky C.
Hetherington and Ramesh Panwar, the disclosures of which applications are
herein incorporated by this reference.

--------- KWIC ---------

Brief Summary Text - BSTX (14):

Using the transit bit, a second access to the same cache line can detect
when a thrash would occur, and either find another tag against which to
reference this access (if available) or stall the processor until a tag becomes
available. By finding another tag, the effect is to allocate a second cache
line to hold the returned data, from the second access to prevent thrashing.
When the processor is stalled, memory access is slowed and overall processor
performance is reduced.